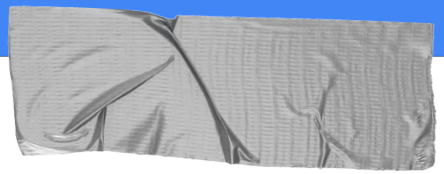
Git and Version Control

Jessica Bell

Women in Tech Summit Northeast 2018

@SirJessTheBrave



Agenda

Git Workshop will take us from the basics of git commands, introduce us to GitHub and give us some hints for when you mess up!

- ➔ **Git Basics**
  Clone, branch, push, and merge
- ➔ **GitHub**
  Git & GitHub, workflows, GUI tools
- ➔ **Git... Oh Shit**
  What happens when it all goes wrong

Jessica Bell

I am a front end developer at The Washington Post, a teacher at General Assembly, and the Producer of DC Tech Stories.

Self taught with a degree in international relations, I learned to code on my own.

@SirJessTheBrave

Git Basics

Git is version control software that you can install in a “repo” (folder where your code files live).

Once installed Git provides commands for programmers to take snapshots (commits) of their code base/repo at a given state (much like track changes).

Vocabulary

- **Repo**
  Folder where your code lives.
- **Init**
  Installing git tracking locally vs. cloning.
- **Commit**
  Taking a snapshot of your code.
  - Stage
  - Commit Message

@SirJessTheBrave

Git Commit

Staging

```
git add
```

Captures your file's state and bundles them with a date and unique ID

Committing

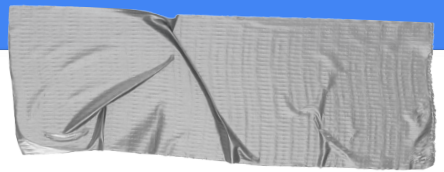
```
git commit -m "my message"
```

Adds a personalized message to your bundled files

Pushing

```
git push
```

Takes the bundle and uploads it to your internet based Git service



Vocabulary

- **Branch**
  A parallel copy of a repo where a developer can work independently.
- **Checkout**
  Switch branches
- **Merge**
  Adding the code written on a branch back into the main codebase.

@SirJessTheBrave

Git Branch

Branch

```
git checkout -b 'newBranch'
```

Creates a new branch by coping the code of the branch you are on

Checkout

```
git checkout newBranch
```

Switches to the code base of an existing branch

Merge

```
git merge develop
```

Merges the branch you state into the branch you're on currently

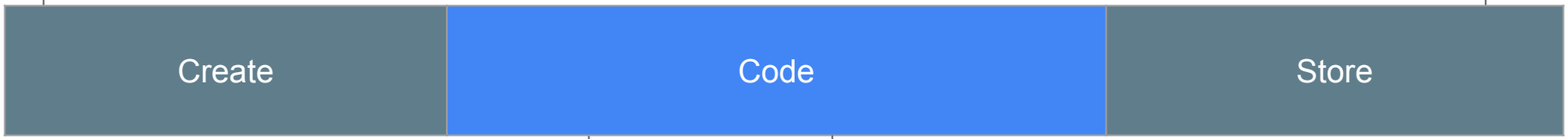
Order of Operations

Init

“Don’t forget to initialize git on your new project”

Commit

“Make sure you commit your work often!”



Optional:
Create or Checkout Branch

“Make a new branch for your feature!”

Optional:
Merge Branches

“When your feature is done merge it into develop”

@SirJessTheBrave

Helpful Commands

Status

```
git s
```

Gives you a status of what is going on in your repo at that moment

Logs

```
git log
```

Gives you the log of recent commits with their messages and ID's

Help

```
git help
```

Gives you a list of commands and their options

https://bit.ly/2qgvAMv

Work Point

Install Git on your machine

Create GitHub Account

Tip

For Mac Os

- Use Homebrew
- `brew install git`

For Windows

- https://gitforwindows.org/

Command Line Vs Graphical User Interface

Command Line vs GUI

Command Line

iTerm for Mac OS

Powershell for Windows

Using your command line to make commands to git

GUI

GitHub

SourceTree

A program you download which provides a click interface to make git commands

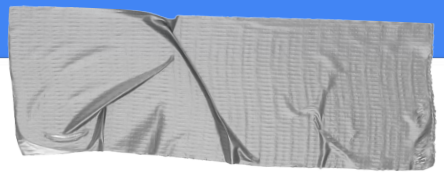
GitHub

GitHub provides cloud storage for your git repositories and collaborative workflow tools such as Pull Requests, merge helpers, and collaborative projects.

GitHub makes it really easy to collaborate

Shows developers a list of branches and gives a graphical user interface (GUI) for making and resolving Pull Requests.

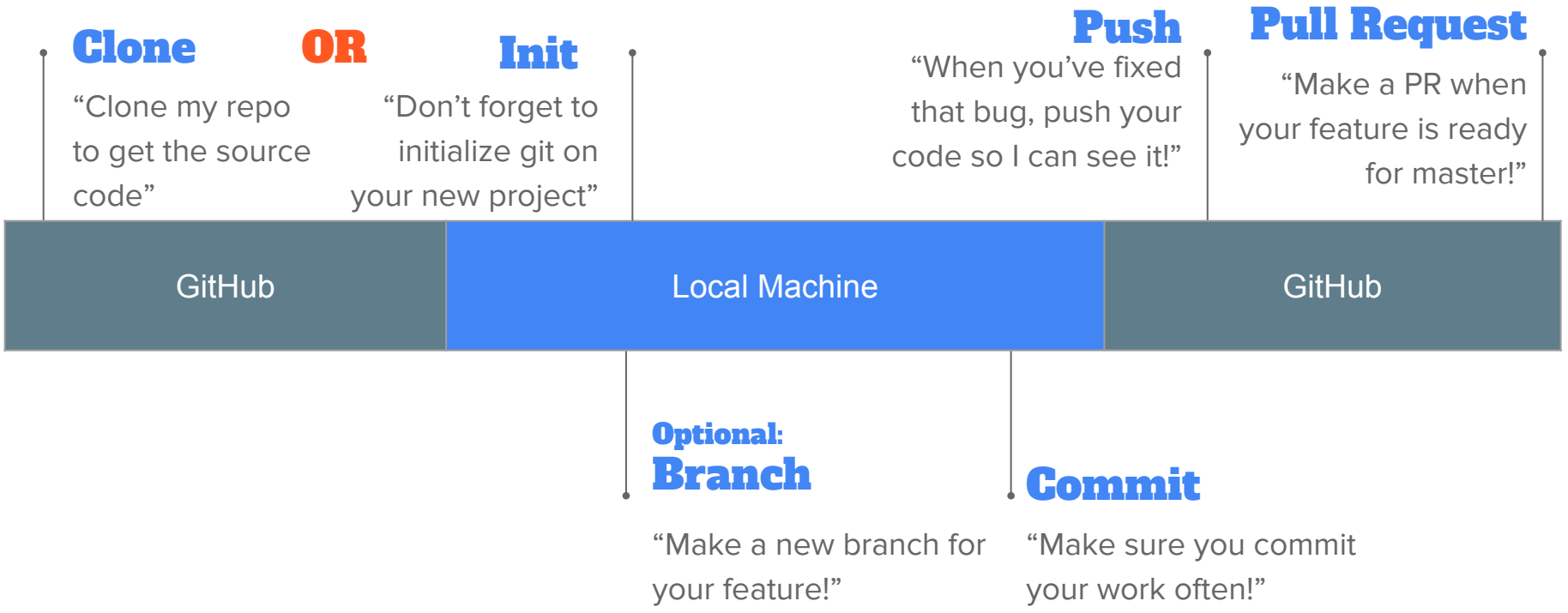
A Pull Request asks to integrate code written by one developer into the main branch.



Vocabulary

- **Clone**
  Downloading a repo from GitHub.
- **Fork**
  Coping a repo from another account.
- **Pull**
  Get the latest code that exists - even what other developers have pushed.
- **Push**
  Upload your latest commit to the internet.
- **Pull Request**
  Request that a branch be merged into another branch.

Order of Operations



Master is always stable

Develop is in flux

Feature for new code

Work Point

Make a new repo locally

Create a README and one initial html file

Push your work to your GitHub account

Hint

~ Git init

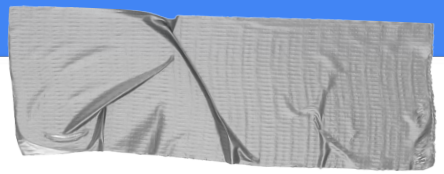
~ Touch README

~ Touch index.html

~ Git add .

~ Git commit -m 'message'

~ Git push



Merge Conflicts

What happens when you do a merge or make a Pull Request that has conflicts

➔ **What**

A conflict means that there is code from both branches on the same line - you need to tell git which one to keep - or both!

➔ **How**

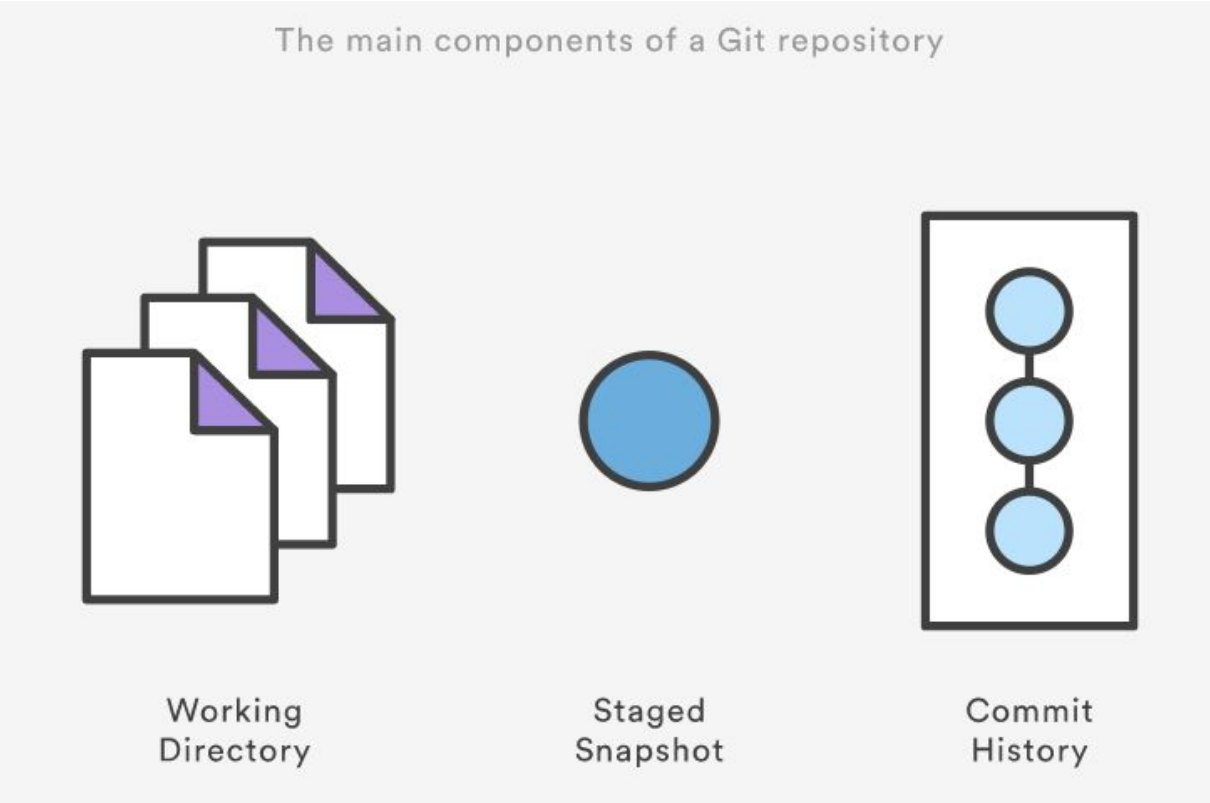
There are a few ways to fix merge conflicts: the command line, your text editor, on GitHub.

Take stock

Are your changes staged or committed?

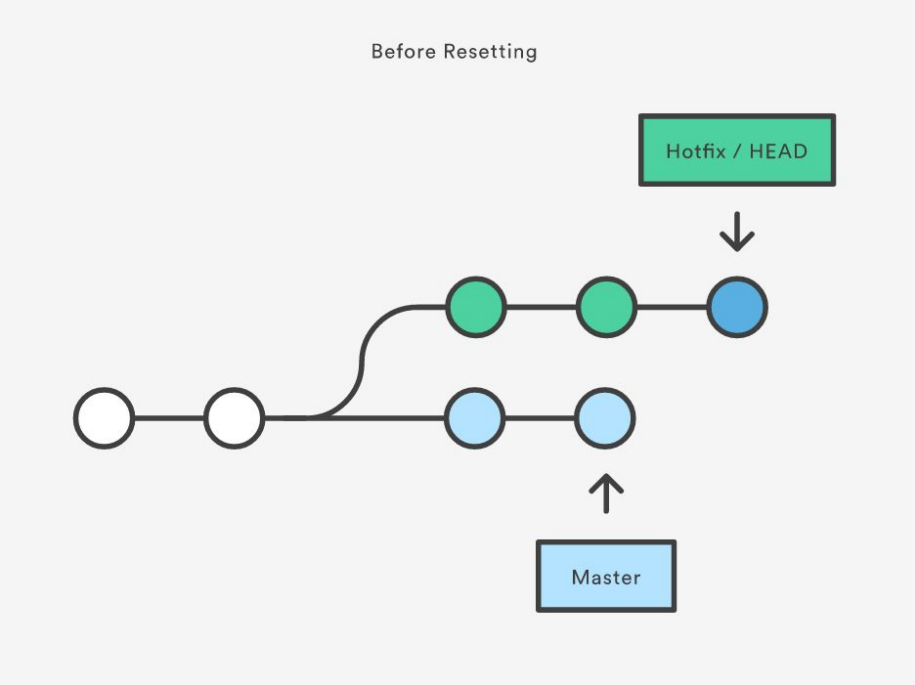
Do you want to preserve or delete the history

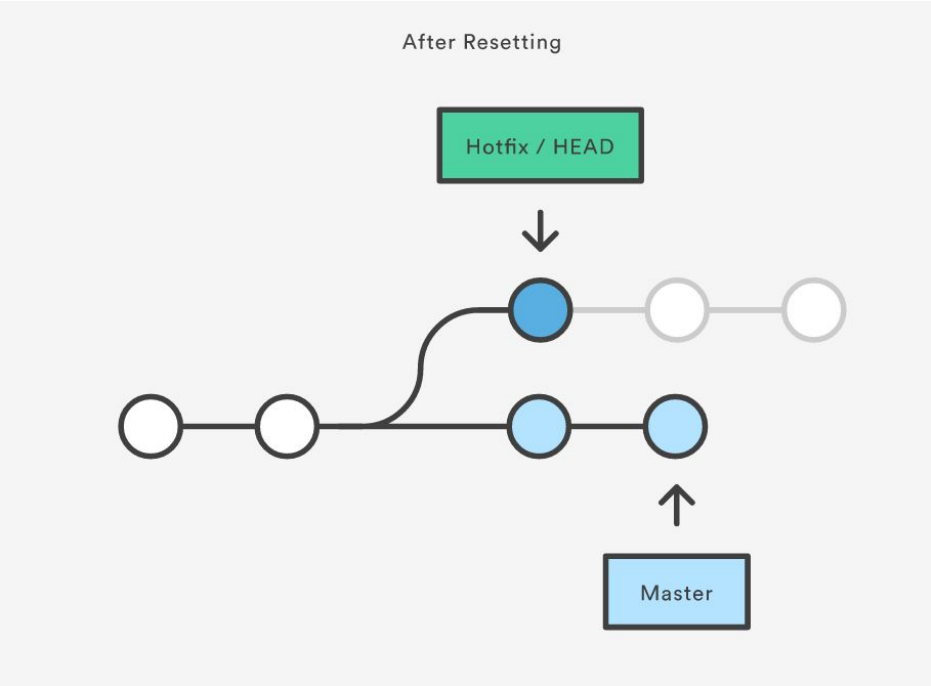
What part of the repo will be effected



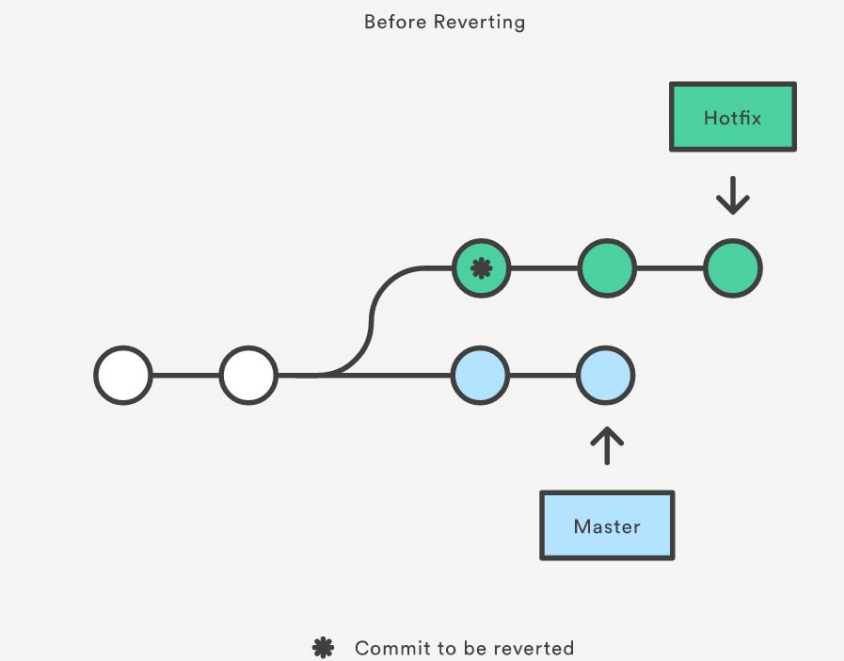
Command	Scope	Common use cases
`git reset`	Commit-level	Discard commits in a private branch or throw away uncommited changes
`git reset`	File-level	Unstage a file
`git checkout`	Commit-level	Switch between branches or inspect old snapshots
`git checkout`	File-level	Discard changes in the working directory
`git revert`	Commit-level	Undo commits in a public branch
`git revert`	File-level	(N/A)

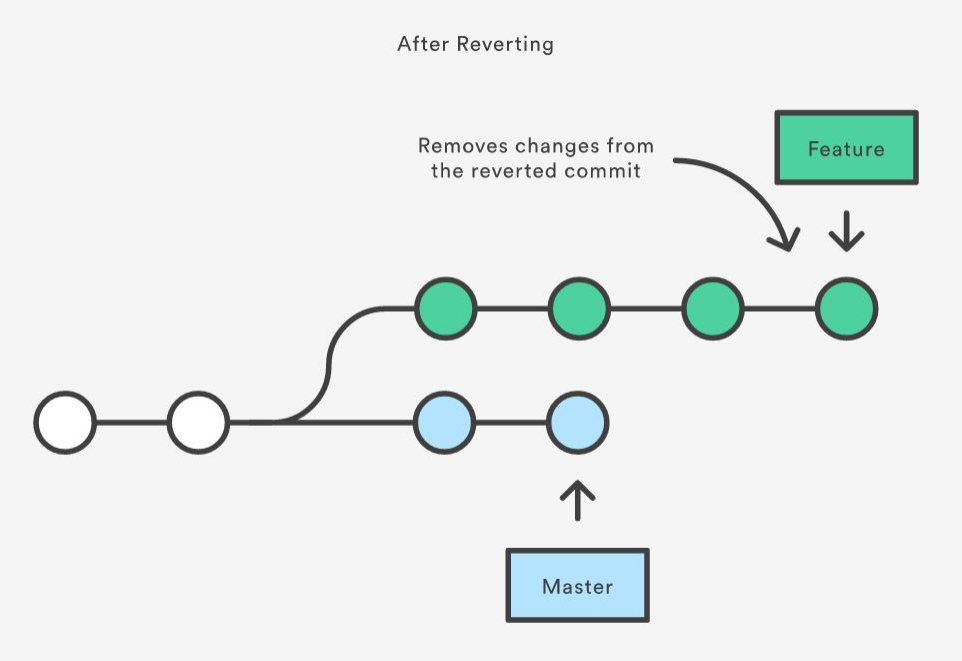
Reset


Before Resetting
Hotfix / HEAD
Master


After Resetting
Hotfix / HEAD
Master

Revert


Before Reverting
Hotfix
Master
Commit to be reverted


After Reverting
Removes changes from the reverted commit
Feature
Master

Level up your Git!

One of the most powerful and awesome things about git is you can roll back your code base to certain moments in time.

One of the most frustrating things about git is HOW exactly to do that.

Diff

See the differences that are conflicting

Tip

Mege is a **non-destructive** operation.

That can pollute the feature branch if the ‘master’ branch is very active.

Stash

Put your changes aside for a while to pull or update code then apply your changes when ready

Tip

Stash is great for when you are coding and want to **pull** code pushed by another developer.

Merge

Takes the one branch and combines the code with another branch

Tip

Mege is a **non-destructive** operation.

That can pollute the feature branch if the 'master' branch is very active.

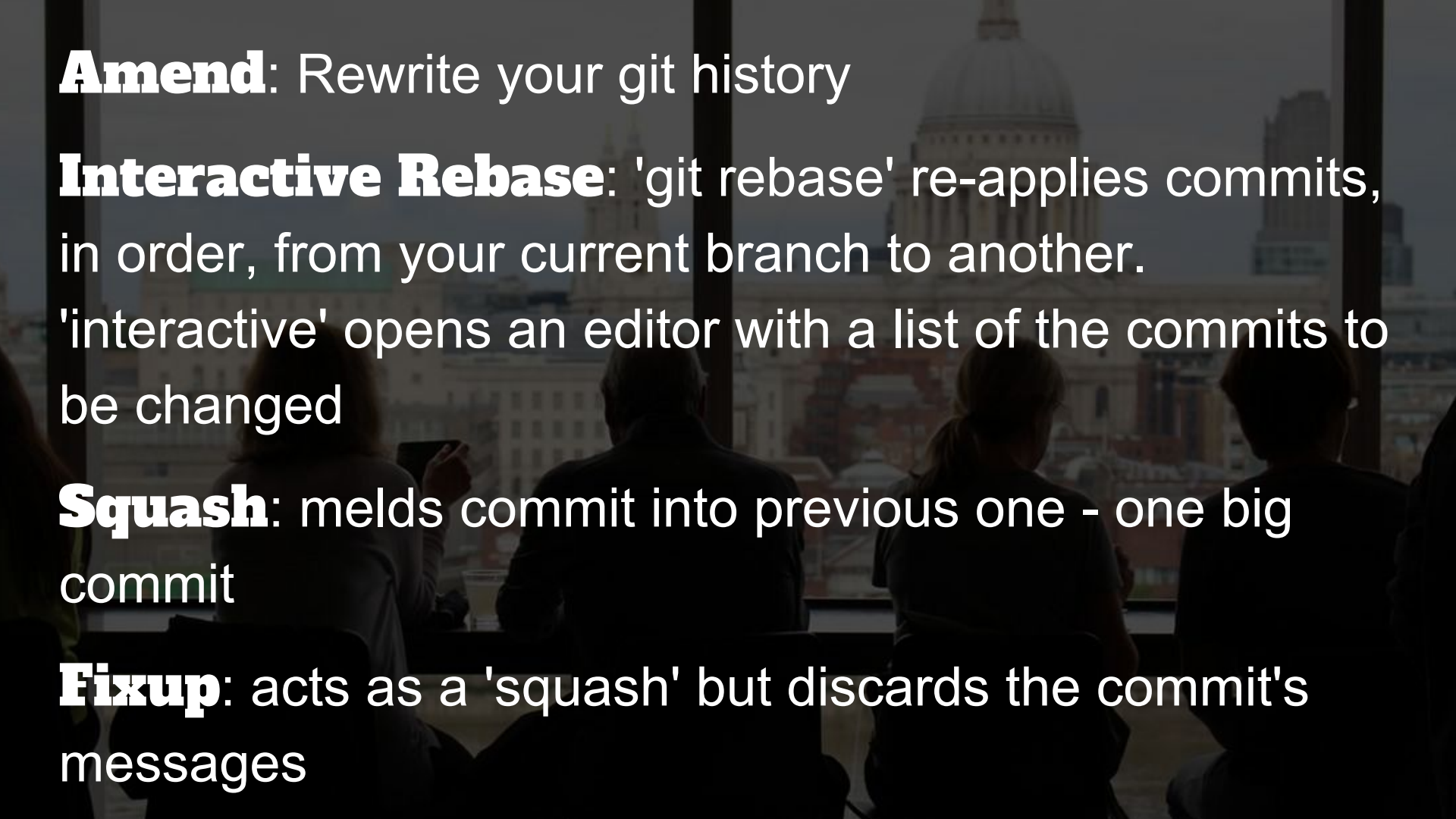
Rebase

Move entire feature branch into second branch - re-writing the project history

Tip

Rebase is a **destructive** operation.

That that means that the history of the branch is re-written.

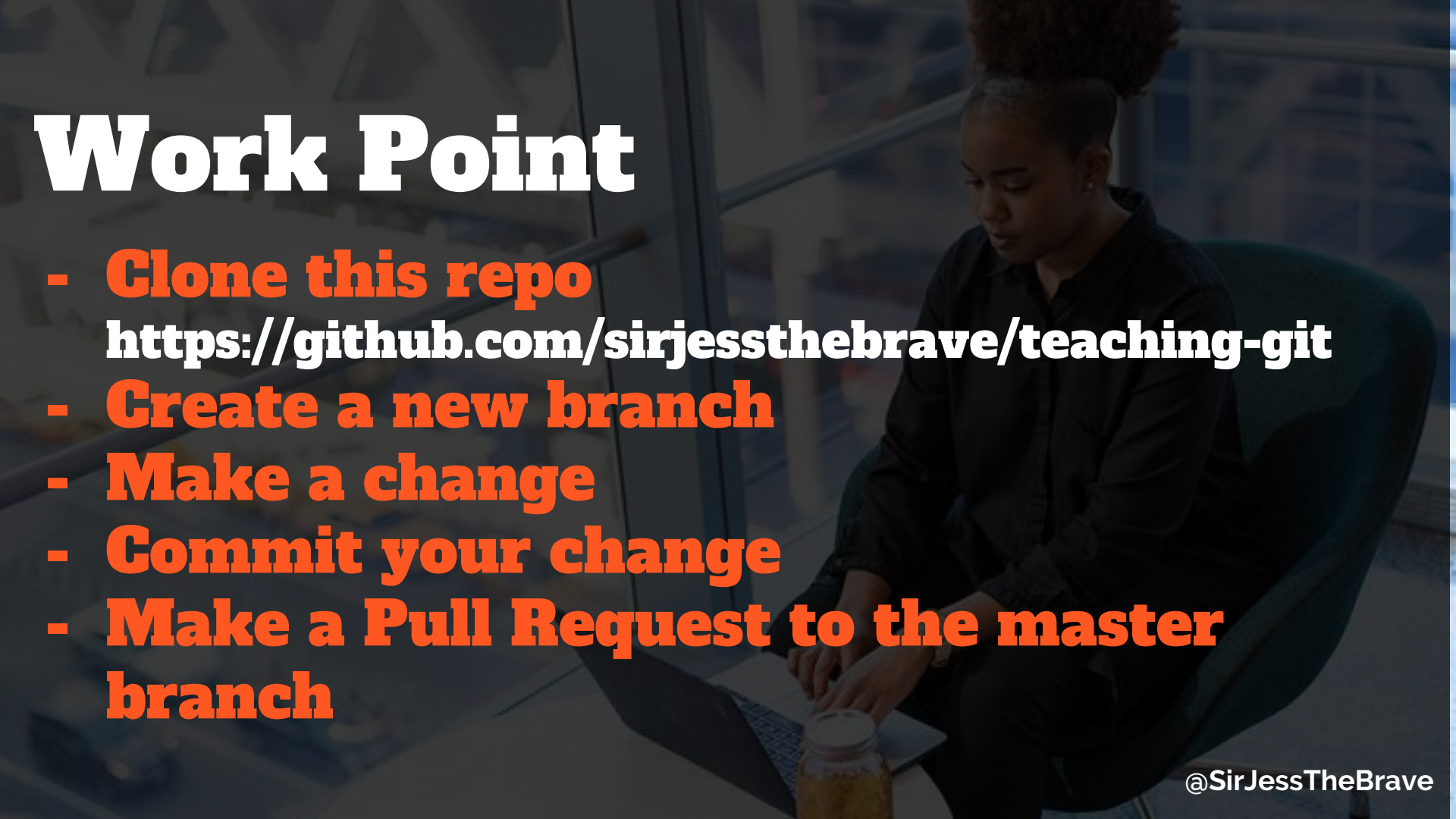


Amend: Rewrite your git history

Interactive Rebase: 'git rebase' re-applies commits, in order, from your current branch to another. 'interactive' opens an editor with a list of the commits to be changed

Squash: melds commit into previous one - one big commit

Fixup: acts as a 'squash' but discards the commit's messages



Work Point

- Clone this repo
  https://github.com/sirjessthebrave/teaching-git
- Create a new branch
- Make a change
- Commit your change
- Make a Pull Request to the master branch

@SirJessTheBrave

Thank You

Jessica Bell

Women in Tech Summit Northeast 2018

@SirJessTheBrave

Reference

- https://www.atlassian.com/git/tutorials/learn-git-with-bitbucket-cloud
- https://www.atlassian.com/git/tutorials/resetting-checking-out-and-reverting
- https://flaviocopes.com/git/#github-desktop
- https://jaimeiniesta.github.io/learn.github.com/p/diff.html
- https://git-scm.com/book/en/v2/Git-Branching-Branching-Workflows
- https://flaviocopes.com/git/
- https://medium.freecodecamp.org/a-developers-introduction-to-github-1034fa55c0db